IDA DOCUMENT D-1111

Ada COMPILER VALIDATION SUPPORT: FISCAL YEAR 1991

R. Danford Lehman
Audrey A. Hook, *Task Leader*

January 1992

Prepared for
Ada Joint Program Office

INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

DEFINITIONS

IDA publishes the following documents to report the results of its work.

Reports

Reports are the most authoritative and most carefully considered products IDA publishes. They normally embody results of major projects which (a) have a direct bearing on decisions affecting major programs, (b) address issues of significant concern to the Executive Branch, the Congress and/or the public, or (c) address issues that have significant economic implications. IDA Reports are reviewed by outside panels of experts to ensure their high quality and relevance to the problems studied, and they are released by the President of IDA.

Group Reports

Group Reports record the findings and results of IDA established working groups and panels composed of senior individuals addressing major issues which otherwise would be the subject of an IDA Report. IDA Group Reports are reviewed by the senior individuals responsible for the project and others as selected by IDA to ensure their high quality and relevance to the problems studied, and are released by the President of IDA.

Papers

Papers, also authoritative and carefully considered products of IDA, address studies that are narrower in scope than those covered in Reports. IDA Papers are reviewed to ensure that they meet the high standards expected of refereed papers in professional journals or formal Agency reports.

Documents

IDA Documents are used for the convenience of the sponsors or the analysts (a) to record substantive work done in quick reaction studies, (b) to record the proceedings of conferences and meetings, (c) to make available preliminary and tentative results of analyses, (d) to record data developed in the course of an investigation, or (e) to forward information that is essentially unanalyzed and unevaluated. The review of IDA Documents is suited to their content and intended use.

The work reported in this document was conducted under contract MDA 903 89 C 0003 for the Department of Defense. The publication of this IDA document does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agency.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1992	Final

4. TITLE AND SUBTITLE
Ada Compiler Validation Support: Fiscal Year 1991

5. FUNDING NUMBERS
MDA 903 89 C 0003

Task Number T-D5-304

6. AUTHOR(S)
R. Danford Lehman

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Institute for Defense Analyses (IDA)
1801 N. Beauregard St.
Alexandria, VA 22311-1772

8. PERFORMING ORGANIZATION REPORT NUMBER
IDA Document D-1111

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Ada Joint Program Office
Room 3E114, The Pentagon
Washington, DC 20301-3081

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, unlimited distribution: ~~30 February 1992.~~

12b. DISTRIBUTION CODE
2A

13. ABSTRACT (Maximum 200 words)

This report summarizes the technical issues that affected the way the Ada Joint Program Office (AJPO) conducted the Ada compiler validation process during fiscal year 1991. These issues are limited to interpretations of Ada language rules, as they are implemented in conformity test programs, and the expected behavior of Ada compilers. The majority of the validation issues that arose pertained to a limited class of language constructs or a particular Ada compiler and hardware and software environment. This report documents IDA's work in fiscal year 1991 addressing the test disputes, validation reports, and compiler registration requests.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Ada Validation; Ada Compiler Registration; Ada Compiler Test Disputes; Ada Compiler Test Reports; Ada Compiler Certification	78
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

IDA DOCUMENT D-1111

Ada COMPILER VALIDATION SUPPORT: FISCAL YEAR 1991

R. Danford Lehman
Audrey A. Hook, *Task Leader*

January 1992

INSTITUTE FOR DEFENSE ANALYSES
Contract MDA 903 89 C 0003
Task T-D5-304

PREFACE

The issue resolution process described in this report was requested by the Ada Joint Program Office (AJPO) as an on-going, integral part of DoD's Ada language control program. The issues resolved during fiscal year 1991 are the subject of this report. These issues relate only to the language conformity test method and their resolution was based upon the analyses of the IDA team consisting of Mr. R. Danforth Lehman (principle analyst), Dr. Cy Ardoin, and Dr. Reginald Meeson. Ms. Audrey Hook, Task Leader, provided oversight and support on policy and procedural issues. This report is for the Director, AJPO and others who need to understand these technical issues as they relate to the Ada test method and the validation process.

Special thanks are due to members of the Fast Reaction Team who assisted the IDA team to resolve complex language and compiler implementation issues: Dr. John B. Goodenough (Software Engineering Institute), Dr. Robert Dewar (New York University), Dr. Norman Cohen (IBM), Dr. Stephan Heilbrunner (Salzburg Univ.), Dr. Erhard Ploedereder (Tartan Labs), Dr. Brian Wichmann (National Physical Laboratory-UK), and Dr. Kenneth Dritz (Argonne Labs). The IDA reviewers of this report were Ms. Kathleen Jordan, Mr. Clyde Roby, Dr. Richard Wexelblat, and Dr. Richard Ivanetich.

EXECUTIVE SUMMARY

The Ada Joint Program Office (AJPO) compiler validation policies and procedures include a formal process for resolving technical issues raised by compiler vendors and testing laboratories and for quality control of the test method. IDA has been tasked by the AJPO to assist in resolving technical issues and to provide independent analyses of the products resulting from the Ada compiler conformity testing process (e.g., test reports and test programs). The resolution process relies heavily on the use of the Defense Data Network (DDN) and IDA's internal network as a mechanism for interacting with Ada language experts who contribute their observations and analyses of specific issues. Resolutions are formulated from interpretive analyses of the Ada Language Standard (ANSI/MIL-STD-1815A), test reports, and previous arguments that set a precedent for resolution. IDA maintains a current and historical data base containing pertinent data about these technical issues and their resolutions. This report summarizes the detail entered into this data base during fiscal year 1991 for the 98 test disputes, 172 validation reports, and 110 compiler registration requests.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 PURPOSE

The Ada Joint Program Office (AJPO) has tasked the Institute for Defense Analyses (IDA) to provide independent analyses of the Ada compiler conformity testing activity and to resolve technical issues involving interpretations of the validation tests or test results. This report partially fulfills the requirements of IDA task T-D5-304, Ada Validation, by providing the AJPO with a summary of IDA activities, that have supported the successful operation of the Ada compiler validation process.

1.2 SCOPE

This report describes the analyses of products resulting from the Ada compiler conformity testing process, analysis of tests for use in the development of the Ada 9X Ada Compiler Validation Capability (ACVC), and analysis of test programs to be used to validate an SQL implementation with Ada bindings.

1.3 BACKGROUND

IDA has been actively involved in the Ada compiler validation process since the first Ada compiler was tested in 1983. Since that time, IDA has provided a range of technical support to the AJPO for the purpose of establishing a certification system based on sound conformity testing practices. The Ada certification system is recognized by the National Institute for Standards and Technology (NIST); within the Ada certification system, IDA is source of technical decisions regarding test issues raised by compiler implementers during the validation process.

The task work includes resolving compiler implementer challenges to ACVC test programs, reviewing each report that documents the result produced from the formal testing process, and recommending to the sponsor that validated status be awarded to specific Ada implementations. Vendors of validated compilers may extend the validated status to maintained compilers by a registration process. IDA analyzes each registration request, and either recommends to the AJPO that the registration be made, or else informs the AVF (Ada Validation Facility) of why the request is refused or what further information is needed. In

performing these tasks, IDA interacts with national and foreign test laboratories that do the Ada compiler validation testing, with Ada language experts of the language maintenance body (the ISO WG 9 Ada Rapporteur Group (ARG)), and with the sponsor.

2. VALIDATION ACTIVITY FOR FY 1991

2.1 ACVC TEST DISPUTES

A "dispute" is defined by the ACVC User's Guide as any result from processing an ACVC test program that is not a passed or inapplicable result according to the established grading criteria. IDA wrote this broad definition in an effort to ensure that implementers brought all deviant test results to the attention of the AVF, without assuming that such results would be accepted without special review. Disputes are forwarded to IDA by the test laboratories, the AVFs, on behalf of their validation customers. IDA has a consulting body of Ada experts, known as the Fast Reaction Team (FRT), who assist in the analysis of test disputes. FRT deliberations are carried out by electronic mail (e-mail). IDA responds directly to the AVF that submitted the dispute. Appendix A presents short summaries of each of the disputes that were received during FY 1991.

For each dispute that is accepted (i.e., when IDA rules in favor of the dispute), it is likely that some correction is indicated for the disputed tests. IDA withdraws any test that is found to be incorrect to a degree that makes it unsuitable for validation. The withdrawal of a test consists of including it on a list of tests whose results are ignored for validation (if they are even processed, which they need not be). IDA updates the list of withdrawn tests, and distributes this list to the AVFs and the AMO (ACVC Maintenance Organization). IDA makes recommendations to the AMO regarding disputed tests.

2.1.1 Dispute Resolution

IDA resolves disputes by any of three methods: a resolution that was made previously is applied to the current dispute (e.g., the same dispute might be submitted at different times by different petitioners); the resolution can be determined unequivocally based on the Ada standard or Ada Commentaries; or, the resolution is determined based on the deliberations of the FRT. Although the Ada Compiler Validation Procedures do not set a maximum or even recommended length of time for reaching a resolution, IDA attempts to resolve disputes within two weeks, an informal guideline that was established by the certification body.

On receipt of a dispute, IDA checks whether the issue matches any that had been previously resolved. If the dispute is new, it is given an initial analysis by IDA; this generally

involves research using the Ada Commentaries in conjunction with the Ada standard and references to previous dispute deliberations. When IDA cannot reach a decision from an examination of the various reference materials, or has questions regarding their interpretation, the dispute is referred to the FRT. IDA presents the dispute and any additional information resulting from IDA's initial analysis to the FRT by e-mail. Deliberation of the dispute proceeds with the exchange of each expert's opinion and analysis. IDA participates in the deliberation by providing information as requested (e.g., ACVC tests or information from the petitioner), eliciting discussion from the experts, and making or challenging technical points raised in the discussion. In general, where an issue receives support from some of the FRT, the dispute is accepted. Most of the disputes that were referred to the FRT during fiscal year 1991 were accepted (15 of 19 cases). Precis of fiscal year 1991 Fast Reaction Notices (FRNs) are attached as Appendix B.

There is no prescribed formality to the FRT deliberations, such as voting procedures or time limits on deliberation. IDA might extend deliberation when a basis for resolving the dispute has not been made. IDA will make its resolution on the dispute when a sufficient basis has been established, regardless of whether the FRT discussion has stopped. An example of FRT deliberation is attached as Appendix C.

2.1.2 Types of Dispute Resolution

The resolution of a dispute is either an acceptance or rejection of the petitioner's arguments. Acceptance can result in either withdrawal of the test program from the ACVC or in a "Test, Processing, or Evaluation" modification for validation. A dispute may be rejected if it conflicts with the Ada standard or Ada Commentaries. Even if the standard and Commentaries do not provide clear grounds for rejection, a dispute may still be rejected if there is no compelling reason to accept it; i.e., an ACVC test program, because it is something to which previously validated implementations conform, is considered to be a sufficient basis for rejecting a dispute in the absence of good arguments for accepting it. A dispute may lead to the withdrawal of a test program if the test is shown to be incorrect to a degree that wrongly influences implementation. Withdrawn tests have no effect on validation (they are generally not processed). If the dispute shows the affected test program(s) to be incorrect in only a minor, limited degree, generally IDA will direct that the test(s) be processed with a test modification.

There are three types of test modification: Test, Processing, and Evaluation modifications. A Test Modification is an actual change to the code of the test (e.g., adding a choice to an exception handler). A Processing Modification is a change to the way in which the test is processed (e.g., re-ordering the compilation of component files of a multi-compilation test). And an Evaluation Modification is simply the grading of the observed results by other than the established grading criteria (e.g., interpreting particular intermediate output and a final "failed" result as "passed", according to an understanding of the dispute). Some examples of the three types of modification are presented in Appendix D.

The major issue to arise during fiscal year 1991 was the vulnerability of many tests to optimization that removes assignments to unused variables. Many of the tests that check that an exception is raised under prescribed conditions use code that does not prohibit the exception-raising expression from being eliminated. The Ada standard (11.6:7) permits an operation to be eliminated "if its only possible effect is to propagate a[n] exception". In many of the tests this is precisely the case, as the expression returns a value that is intended to be assigned to a variable that is never used (hence, the value is not needed and the assignment and expression evaluation need not be made). IDA rejected only one case of optimization, where it removed a programming safeguard (local variable initialization for a block) and was too far from what the standard clearly permits.

2.1.3 Dispute Resolution Workload

In fiscal year 1991, AVFs submitted over 150 disputes; 98 of these were unique disputes (these are presented in Appendix A), 44 of which were new. The average response time for the resolution of a new dispute was 11 days, with a standard deviation of 9 days. IDA rejected 15 disputes and issued modifications for 74 others. Nine ACVC tests were withdrawn as a result of disputes.

2.2 REVIEW OF ADA COMPILER VALIDATION SUMMARY REPORTS (VSRs)

For each Ada implementation tested, an AVF produces a Validation Summary Report (VSR). This VSR contains a general indication of the validation test results: it describes why inapplicable tests were considered to be inapplicable, and it describes any modifications made to the test programs for validation. In an annex, the VSR lists the set of values used in the customization of the ACVC (these are values designed to be inserted into certain

tests by a customizing program). Additional annexes contain vendor-supplied information about the implementation. VSRs are often produced in advance of validation testing, using pre-validation results (from vendor self-testing) as the basis for the sections on inapplicable and modified tests—AVF testing is intended to be a pro forma verification of the vendor-submitted results. IDA is tasked with ensuring that AVFs produce accurate reports of validation testing. IDA keeps copies of all final VSRs, which serve as an information base for the validation task and other tasks.

AVFs produce the VSR from a template which contains all of the unchanging text common to different VSRs as well as a set of descriptions of commonly inapplicable or modified test programs. It is the second chapter of the VSR that provides the most important, implementation-specific information about validation testing: in separate sections, it lists the names of the withdrawn, inapplicable, and modified test programs. The sections on inapplicable and modified test programs describe why a test program was inapplicable or why and how it was modified. An AVF simply selects the appropriate pre-written entries for these two sections of the VSR. Where existing entries are not appropriate, IDA works with the AVF to write a new entry.

When more than one implementation is tested during a single validation effort by a customer (e.g., 286/DOS, 386/UNIX, Macintosh, & Sun-3/SunOS), an AVF may submit a single complete draft VSR and show differences for the other validations in VSR "stubs". A VSR stub is an abbreviated VSR that includes only information that differs from a given complete VSR; for example, one implementation might have additional inapplicable or modified test programs, thus the pages on which these were described would be part of the stub. By producing only stubs, differences are highlighted, paper and mailing costs are reduced, and the effort to review the entire set of VSRs for the group validation effort is reduced considerably. In many cases the result profiles for implementations are nearly the same, and VSR stubs thus carry only implementation information (possibly only the different computer system designations).

2.2.1 VSR Review

IDA reviews each VSR for plausibility: because the actual results are not indicated except by the reporting action of the AVF, IDA can only determine whether the report describes results that are plausible and acceptable The VSR review is conducted according to a review checklist which serves to guide the review and ensure that all important items

are checked. Comments on the VSR, i.e., recommended changes to be made, are sent by e-mail to the particular AVF. Because the draft VSR is reviewed in detail, the final VSR, which will be signed by the AVF, IDA, and the AJPO, is reviewed principally to see that all comments are reflected in the VSR. At this stage, if minor errors remain, IDA will either make the correction or else simply ignore the error (e.g., a misspelling might be ignored, or extraneous text might be deleted). It might be the case, however, that further comments will be sent to the AVF so that corrected pages can be produced and incorporated.

2.2.2 VSR Workload

IDA's FY91 workload for the review and handling of VSRs is indicated by the following data:

a. draft VSRs received: 187
b. comments on drafts: 172
c. final VSRs received: 152
d. final VSRs reviewed: 139
e. comments on finals: 51
f. final VSRs delivered: 109

What is meant by "comments" in items "b" and "e" above are the e-mail messages from IDA to the AVFs (and not individual comments): e.g., IDA made comments for 51 final VSRs. As for the number of particular comments contained in each e-mail message, only a coarse indication is given here, based on a review of the e-mail sent for 150 VSRs: on average, four particular comments were made for each validation (i.e., for both draft and final VSRs, where applicable). This figure is "coarse" because it is based simply on the numbered particular comments of each e-mail message. No attempt has been made to examine all of these comments in detail to determine the number of points each addresses (e.g., a particular comment might provide several revised VSR Section 2.3 entries).

In addition to the review work indicated by the data above, IDA also revised the VSR template by rewriting the individual entries for Section 2.2 (Inapplicable Tests) and creating standard entries for Section 2.3 (Test, Processing, and Evaluation Modifications). IDA continues to write all new Section 2.3 entries as needed.

2.3 VALIDATION CERTIFICATE REQUESTS

IDA initiates the issuance of a validation certificate for AVF customers by sending a certificate request to the Ada Information Clearinghouse (AIC), one of the sponsor's functions which is supported by a contractor on site. The certificate request is completed based on information presented in the VSR and in a signed statement from the AVF customer called a Declaration of Conformance (DoC). IDA makes a certificate request to the AJPO only after reviewing the VSR and obtaining the exact information needed for the certificate. IDA requested 186 certificates during fiscal year 1991.

2.4 REGISTRATION REQUESTS

Registration is a process whereby a vendor may extend the scope of a validation certificate to include closely related Ada implementations. Typical registrations, for example, are those that are for a base compiler in operation on other members of the tested configuration's host and target computer families, or for an upgraded (maintained) version of the compiler. To register an Ada implementation, a vendor must establish that the implementations stand in an acceptable relation to what was tested; i.e., the vendor must establish that the computer systems have the same or compatible instruction sets and operating systems, and that the compiler is the same or has been changed only within the scope of software maintenance. Additionally, a vendor must declare that the results of processing the ACVC test suite are or will be the same as those established during validation testing (minor differences may be accepted depending on the explanation for them). Finally, a vendor must submit a signed DoC for the registered implementation.

2.4.1 Registration Workload

IDA received 110 registration requests during fiscal year 1991; only 6 of these were refused, due to the failure of the customer to provide either complete or correct information. IDA prepares the registration information for presentation on the Validated Compilers List; such preparation entails occasional editing of the computer-system designations, grouping the registered implementations according to computer systems or operating system versions, or directing the AIC to edit an existing listing to include the new information.

2.5 STATUS REPORTS

IDA tracks the status of each validation effort for which a draft VSR or DoC has been received. At the close of each week, IDA issues a status report which shows the dates of various actions (e.g., the receipt of the draft VSR, the issuance of comments or a certificate request). Validations are reported on until all actions are completed: i.e., after the report shows the date of IDA's delivery of the final VSR to the AJPO, the entry for that validation is removed.

2.6 CONTRIBUTIONS TO ACVC QUALITY CONTROL

At the beginning of FY 1991, IDA contributed to the production of ACVC 1.12 by submitting analyses of 29 problems that affected ACVC test programs to the AMO. IDA also rewrote two test programs, one of which was incorporated into ACVC 1.12.

IDA submitted to the Ada 9X ACVC Reviewers and ACVC Team analyses of test deficiencies for 40 particular ACVC tests as well as critical comments and suggestions for several groups of tests. This analysis was made in the course of performing other tasks (reviewing VSRs and resolving test disputes), and will continue during development of the Ada 9X ACVC.

2.7 SQL-Ada LANGUAGE INTERPRETATIONS

In May 1991, NIST issued a public solicitation for Ada experts: NIST sought help to address Ada language issues that would arise in the course of NIST's development of a test suite for SQL-Ada binding. With the sponsor's approval, IDA offered its Ada expertise to NIST. IDA's participation in NIST's SQL standardization process draws on IDA's prior work, performed under a task that was expressly for development of the SQL-Ada binding. Thus, IDA brings not only Ada expertise resources but also familiarity with the SQL-Ada binding to this new work.

Four language issues were resolved. Three of these issues were simple matters of applying the Ada standard (and, in one case, an Ada Commentary); the remaining issue involved producing technical guidance on whether and how SQL implementations should automatically generate an Ada package as part of the SQL-Ada interface.

3. LIST OF TERMS AND ABBREVIATIONS

ACVC (Ada Compiler Validation Capability): The Ada certification system's means for testing Ada implementations for conformity to the Ada standard. The ACVC consists of a test suite & support programs, a user's guide, and a test-report template (see VSR below).

AIC: Ada Information Clearinghouse.

AJPO (Ada Joint Program Office): The AJPO provides policy and guidance for the Ada certification system.

AMO (ACVC Maintenance Organization): The organization that is responsible for the maintenance of the ACVC.

ARG: Ada Rapporteur Group.

AVF (Ada Validation Facility): A test laboratory that conducts the Ada validation tests for the Ada certification system. Since 1983, IDA has acted as the AVO.

AVO (Ada Validation Organization): The source of technical guidance for the operations of the Ada certification system.

Ada certification system: The system, with its rules of procedure and management as defined by the AJPO, for carrying out Ada compiler conformity certifications.

Ada certification body: The AJPO, the AVO, the AVFs, and the AMO, who operate the Ada certification system.

Ada implementation: An Ada compiler together with its host and target computer systems.

Ada standard: The standard for the Ada programming language, which is available as ANSI/MIL-STD-1815A-1983, ISO/8652-1987, & FIPS 119, 1985.

Base implementation: An Ada implementation that is validated by AVF testing.

DoC (Declaration of Conformance): A formal statement from an AVF customer assuring that conformity to the Ada standard is realized or attainable on the Ada implementation(s) for which validated status is requested.

Dispute: For the purposes of validation, a "dispute" is constituted by any behavior of the candidate Ada implementation that is not explicitly permitted by the code of the ACVC tests, by the documentation from the AMO that accompanies the ACVC tapes, or by other documentation provided to the vendor by the AVF.

FRN: Fast Reaction Notice.

FRT: Fast Reaction Team.

IDA: Institute for Defense Analyses.

ISO: International Organization for Standardization.

NIST: National Institute for Standards and Technology.

Validation: The process of checking an Ada implementation for conformity to the Ada standard.

VSR (Validation Summary Report): A report produced by an AVF containing results that are observed from testing a specific Ada implementation.

WG: Working Group.

Appendix A.

ACVC TEST DISPUTES IN FISCAL YEAR 1991

This appendix lists the disputes for ACVC 1.11 that were adjudicated by IDA during fiscal year 1991. These disputes are presented in the following format:

<resolution indicator> <list of tests affected by this dispute>

<brief description of the dispute>

The disputes are presented in the order of the Ada standard sections to which the they pertain. Each dispute listing is accompanied by the names of the affected tests, a brief description of the dispute, and an indicator of what the resolution of the dispute was. In a few cases, very similar disputes are presented together, with respective descriptions given in separate paragraphs. The resolution indicators are the following, listed in order of their frequencies (given in parenthesis):

EM=NA	(27)	graded inapplicable by Evaluation Modification
TM=PS	(24)	graded passed by Test Modification
EM=PS	(16)	graded passed by Evaluation Modification
RJ***	(15)	the dispute was rejected
WD---	(9)	withdrawn from the ACVC (i.e. ignored for validation)
PM=PS	(8)	graded passed by Processing Modification
Q?...	(3)	the dispute was questioned and then dropped

There is also one "NOTE" (viz. #86, re CE3602A). This isn't a "dispute" in the sense that there was some deviant implementation behavior, but it did involve AVO analysis which resulted in a recommended correction being made to the ACVC Team. Note that disputes might have more than one resolution indicator, because either slightly different circumstances affected the resolution (e.g., see #13, re A35801E), similar disputes are presented, or the affected tests could not all be handled in the same way (e.g. see #09); hence, the sum of the frequencies above is greater than the number of disputes listed below (102 vs. 88).

In describing the disputes, references are made to the Ada standard, the Ada Commentaries, and to particular ACVC tests; these references use the following forms, respectively:

Standard <chapter>.<section>.<subsection>:<paragraph> (e.g. "Standard 3.5:4"),

AI-<number>/<version> (e.g. "AI-00301/07"), and

"<test_name>" (e.g., "C52008B", "C85006A..E"--a series of 5 tests, C85006A through C85006E).

Also, the text often refers to an ACVC report procedure, viz. Report.Failed; this procedure is invoked from executable tests when a check is failed.

CHAPTER 1, INTRODUCTION (ERROR DETECTION; PROGRAM REJECTION)

---------#1----------

Q?... C43204G, C95067A

EM=NA C85005C, C85006C

These tests were challenged by one implementer on the grounds that they exceeded the compiler's capacity—that they required a jump of excessive range in the generated code. After AVO questioning, the dispute of two of the tests was dropped; tests C85005C & C85006C were allowed to be graded inapplicable.

---------#02----------

TM=PS C85006A..E

These tests caused capacity problems for some implementations (compilation), and were allowed to be split into a set of smaller tests (one implementation used two split versions, another used five).

---------#03----------

EM=PS B74301A

This test checks that an deferred constant declaration given in the visible part of a package is illegal if no corresponding full declaration is in the package declaration; the test expects an error to be given at the end of the private part. Some implementations place the error flag at the point of the incomplete declaration (which is marked as "OK"), and the AVO ruled that this was acceptable behavior and the test did not need to be split to isolate the case.

---------#04----------

EM=PS B83E01B

This test checks that a generic subprogram's formal parameter names (i.e., both generic and subprogram formal parameter names) must be distinct; the duplicated names within the

subprogram declarations are marked as errors, whereas their recurrences in the subprogram bodies are marked as "optional" errors—except for the case at line 122, which is marked as an "error". Some implementations do not additionally flag the errors in the bodies and thus the expected error at line 122 is not flagged. The AVO ruled that such behavior is acceptable and that the test need not be split (such a split would simply duplicate the case in B83E01A at line 15).

--------#05----------

TM=PS C52008B

RJ***

This test uses a record type with discriminants with defaults that has array components whose length depends on the value of a discriminant of type INTEGER. For some implementations, the elaboration of the type declaration raises either NUMERIC_ERROR or CONSTRAINT_ERROR because the attempt to calculate the maximum possible size for objects of the type overflows. (Raising STORAGE_ERROR on the elaboration of the declaration of an object of this type is also permissible.)

For another implementation, the compilation of the type declaration fails when the compiler makes the same maximum-size calculation that is described above. Although this behavior is strictly illegal, it was allowed because the error was recognized only after on-site testing had been completed.

--------#06----------

RJ*** B85002A

EM=PS [this became the final Ruling]

This test declares a record type REC2 whose sole component is of an unconstrained record type with a size in excess of 2**32 bytes; some implementations reject the declaration of REC2. Although a strict interpretation of the Standard requires that this type declaration be accepted (an exception may be raised on the elaboration of the type or an object declaration), the AVO accepted this behavior in consideration that such early error detection was expected to be allowed by the revised language standard. (That expectation is contradicted by the August 1991 Ada 9X Mapping document.)

--------#07----------

WD--- C74308A

One petitioner argued that this test's use of a deferred constant prior to its full declaration made the test erroneous and that the test did not account for all allowable behavior. The FRT concurred in and expanded the petitioner's position. The AVO found other flaws in the test as well.

--------#08----------

TM=PS CC3126A

One petitioner re-raised an ACVC 1.10 dispute of this test's use of an undefined array value. The AVO ruled that the test be modified so as to provide initialization for the array; apparently, validation ultimately did not require this modification.

CHAPTER 2, LEXICAL ELEMENTS

--------#09----------

EM=NA B22005A..C/I/P

TM=PS B25002A, B26005A

EM=PS B25002A, B26005A, B27005A

These tests include checks on the handling of control characters. For some implementations, certain of the control characters have special significance for the underlying computer system such that the tests cannot be processed. For two of the tests, it is possible to accommodate the unusual implementation behavior by commenting out a few lines; in other cases, the evaluation of the results was modified to ignore the lines that checked the following particular control characters: SOH (ctrl-A), STX (ctrl-B), ETX (ctrl-C), NUL (ctrl-@), & DLE (ctrl-P).

--------#10----------

EM=NA B23003D/E

EM=PS B23003F

These tests assume that an implementation imposes a limit on the length of the input line; this implementation has no such limit. The AVO ruled that this behavior is acceptable. As a consequence, B23003D & B23003E are inapplicable, and the B23003F may be interpreted as checking the implementation's limit on identifiers (which all implementations thus far have had).

--------#11----------

RJ*** E28002B, E28005D

TM=PS E28002B

These tests include checks that pragmas LIST and PAGE have the required effects. For some implementations, pragma LIST has effect only if the compilation results has no errors or warnings, which is not the case when E28002B is processed without modification. For these implementations, E28002B was also processed with the pragmas at lines 46, 58, 70, and 71 commented out so that there were no warnings and pragma LIST had effect. One petitioner challenged the need to support the two pragmas; the AVO ruled that they must be supported, as required by Commentary AI-00570.

CHAPTER 3, DECLARATIONS AND TYPES

--------#12----------

WD--- C35702A/B

These tests check that SHORT_FLOAT and LONG_FLOAT have different precision than FLOAT, but commentary AI-00459/08 reverses the Standard and allows the definitions of the different types to be identical.

--------#13----------

EM=NA A35801E

RJ***

A compiler must reject the use of the range FLOAT'FIRST .. FLOAT'LAST as the range constraint of a floating-point type declaration if the bounds lie outside of the range of safe numbers (Standard 3.5.7:12); this test uses such a range constraint but expects it to be legal always. The AVO ruled the test to be inapplicable to implementations with no base type with a larger range than FLOAT.

Some disputes were for Ada implementations with a base type with a larger range than that of type FLOAT; these disputes were rejected.

--------#14----------

RJ*** C35A04D/Q, C35A07D/Q

One implementer challenged these tests on the basis of AI-00144 and the fact that a specified bound of a fixed-point type's range need not be representable; but when the AVO

asked for implementation information, the dispute was dropped. (The AVO analyzed the tests and found them to be correct; perhaps the AVO request helped the petitioner also to see the tests' correctness.)

---------#15----------

TM=PS C45232A

This test contains the expression "INTEGER'LAST > SMALL_INT'BASE'LAST" at lines 131 & 169; the test does not anticipate that if the condition is false, the implicit conversion of the right operand to type INTEGER may raise an exception. One implementation selected type LONG_INTEGER for the base type of SMALL_INT, and so raised an exception. The test was modified by inserting the code 'FALSE THEN --' immediately after 'IF' in both lines, which avoids the exception and accurately reflects the actual condition.

---------#16----------

TM=PS CD2A83A, CD2A84A, CD2A84E, CD2A84I, CD2B11A, CD2B11B

These tests check the use of access types whose type size and collection size have been specified with length clauses. In order to accommodate the Rational R1000 implementation's unusual SYSTEM.STORAGE_UNIT value (1) and requirement that, for any access type T, T'Storage_Size <= 2 ** T'Size, these tests' specified values for access type size and collection size were modified.

---------#17----------

EM=NA BD2A85A/B

These tests each use a length clause to specify the size for an access type, and they expect that the length clause will be rejected by the compiler because the specified size is too small. However, the Rational R1000 implementation treats access values as offsets into the collection, and thus accepts these clauses.

CHAPTER 4, NAMES AND EXPRESSIONS

---------#18----------

TM=PS C34003A

This test checks operations on derived floating-point types and uses a complex expression whose intermediate results caused an overflow for one MIL-STD-1750A implementation. The two expressions at lines 219 & 226 equated to the calculation 2**126,

and this calculation overflowed as a consequence of the way multiplication occurs (even though the result can be represented); NUMERIC_ERROR was raised. Thus, lines 219 & 225 were modified to use an equivalent expression that avoids the intermediate value 2**126.

--------#19----------

TM=PS C45524A..<?> [# of disputed tests depends on supported precision]

These tests expect that a repeated division will result in zero; but the Standard only requires that the result lie in the smallest safe interval. Thus, the tests were modified to check that the result was within the smallest safe interval, by adding the following code immediately following line 141: 'ELSIF VAL <= F'SAFE_SMALL THEN COMMENT ("UNDERFLOW IS GRADUAL"); '.

--------#20----------

WD--- C45612A..C

These tests check that exponentiation with exponents near INTEGER'LAST works correctly (hence, the exponentiated values must be -1, 0, & 1). Petitioners challenged these tests on the grounds that normally efficient exponentiation would consume too much time for such large powers, and that the tests therefore encouraged a degradation of implementations in order to handle these special cases. The FRT concurred in the petitioners' opinion.

--------#21----------

E.=NA C64103A, C95084A

If an implementation's LONG_FLOAT'SAFE_LARGE and SHORT_FLOAT'LAST lie within one (SHORT_FLOAT) model interval of each other, a floating-point applicability check may evaluate to TRUE and yet the subsequent expected exception need not be raised and the tests will report FAILED. The FRT agreed that this behavior is acceptable. The AVO ruled that the tests should be graded as passed because the implementation passed the integer and fixed-point checks.

--------#22----------

EM=PS CE3804H

This test requires that the string "-3.525" can be read from a file using FLOAT_IO and that an equality comparison with the numeric literal '-3.525' will evaluate to TRUE; however, because -3.525 is not a model number, this comparison may evaluate to FALSE

(Standard 4.9:12). Because some implementations' compile-time and run-time evaluation algorithms differ, the check for equality may fail. The test was graded as passed if the only Report.Failed output is from line 81, the message "WIDTH CHARACTERS NOT READ".

--------#23----------

WD--- B49008A

The petitioner showed that this test contradicts Commentary AI-438/09 in that one of the intended illegal lines is held to be legal by the ARG.

CHAPTER 5, STATEMENTS

--------#24----------

RJ*** C54A13D

The petitioner challenged this test's check that choices in case-statement alternatives represent (once and only once) each value of the (base) type of the expression, and not merely those of the subtype (except for some particular circumstances as specified by the Standard 5.4:4). As the challenged case was not among those required by the Standard to be restricted to the subtype, this dispute was rejected. (But the issue was referred to the Ada 9X Mapping Team.)

CHAPTER 6, SUBPROGRAMS

--------#25----------

PM=PS EA3004D

The test requires that either pragma INLINE is obeyed for a function call in each of three contexts and that thus three library units are made obsolete by the re-compilation of the inlined function's body, or else the pragma is ignored completely. Some implementations obey the pragma except when the call is within the package specification. When the test's files are processed in the given order, only two units are made obsolete; thus, the expected error at line 27 of file EA3004D6M is not valid and is not flagged. To confirm that indeed the pragma is not obeyed in this one case, the test was also processed with the files re-ordered so that the re-compilation follows only the package declaration (and thus the other library units will not be made obsolete, as they are compiled later); a "NOT APPLICABLE" result was produced, as expected. The revised order of files was 0-1-4-5-2-3-6.

One petitioner made statements regarding an implementation's behavior on this test that lead to an full question-and-answer exchange; it was eventually concluded that, since the implementation could process the test correctly as per the guidance above, the issue should not be pursued further.

CHAPTER 9, TASKS

---------#26----------

PM=PS C32107A, C34007J, C64201C, C85006A..E, C93004C..D, C93005B..D, C93005F, C93005H, C94001B, C94002A, C94007A..B, C95021A, C95022B, C95066A, C95071A, C95087A, C97307A, A98002A, C99005A, C9A008A, CD2A91A..B, CD2A91E

The petitioner argued that the implementation must process these tests with special settings for the primary and task stacks in order to avoid raising STORAGE_ERROR. The FRT accepted the dispute, but ultimately only C32107A was processed with special stack sizes, due to changes to the environment.

---------#27----------

TM=PS C64201C, A98002A, C99005C

Q?... A98002A

These tests respectively contain 12, 17, & 12 tasks. One implementation used a default amount of storage for tasks that led to STORAGE_ERROR being raised. For this implementation, these tests were modified to include length clauses that specified 1K bytes for the task storage size (for tests C99005A & A98002A, this required that the single tasks be re-written as task types).

Another implementation relied on the use of an operating system's threads to implement tasks, and could not execute A98002A because the OS did not provide enough threads. The petition for this implementation was dropped before a ruling was made; an earlier version of the OS offered more threads, and the problem did not arise when that was used during validation.

---------#28----------

TM=PS C94020A, C95020D

These tests have output from several tasks which can become intermixed when printed. Package Report was modified so as to enclose the output procedure within a task so that the

tests' tasks' calls to this procedure are handled sequentially instead of in parallel. (Without this modification, it is hard to read the tests' results.)

CHAPTER 10, PROGRAM STRUCTURE AND COMPILATION ISSUES

--------#29----------

TM=PS C83030C, C86007A

These tests may be modified by inserting "PRAGMA ELABORATE (REPORT);" before the package declarations at lines 13 and 11, respectively. Without the pragma, the packages may be elaborated prior to package Report's body, and thus the packages' calls to function Report.Ident_Int at lines 14 and 13, respectively, will raise PROGRAM_ERROR.

--------#30----------

EM=NA B83E01F, BA1011C

These tests expect that the bodies of generic subprograms can be compiled separately from their declarations. One implementation requires that generic declarations and bodies be in the same compilation, which is allowed by the Standard 10.3:9.

--------#31----------

EM=PS BA2001E

The test checks that subunits with a common ancestor cannot have the same name; it expects errors to be detected at compile time. Some implementations detect the errors at link time, and the FRT agreed that this is acceptable.

--------#32----------

EM=NA CA2009A/C/D/F [not all tests are affected by all disputes]

These tests contain instantiations prior to the compilation of the body of the instantiated generic unit. As allowed by AI-00408 and AI-00506, some implementations create a dependence on of the unit that contains the instantiations on the generic units such that the compilation of the generic unit bodies makes the compilation unit that contains the instantiations obsolete.

In some other implementations, the generic bodies are required to be in the same compilation as their specification if the instantiations are compiled before the bodies, as allowed by AI-00257.

--------#33----------

EM=PS BA3001A

This test contains a generic subprogram declaration with no corresponding body. One implementation requires that generic declarations and bodies be in the same compilation, therefore it detected the absence of a subprogram body as an additional error. The AVO ruled that the additional error message may be ignored.

--------#34----------

PM=PS LA3004A/B

These tests check that when the bodies of three library units (a procedure, function, and package) are made obsolete the implementation will detect those missing bodies at link time. Some implementations did detect the missing bodies, but also issued error messages that indicate that the main procedures must be re-compiled; requiring re-compilation would violate the Standard 10.3:6 & 8. To confirm that the implementations do not in fact require this re-compilation of the main procedures, the obsolete bodies were re-compiled (files LA3004A2..4 and LA3004B2..4 were modified to contain only the bodies) and the tests were then linked and executed; then the Report.Result output was "NOT APPLICABLE", as expected.

--------#35----------

RJ*** CC1305B, BC3204B, BC3205B

A petitioner (implicitly) argued that these tests may legitimately fail to compile due to an instantiation preceding the compilation of the respective generic body. However, the FRT confirmed that Commentary AI-00506 granted that allowance only in cases where the generic body was separately compiled, and the AVO rejected the dispute.

--------#36----------

EM=NA BC3009C

This test checks that circular instantiations of two and three levels are detected as illegal. One implementation rejected the package specification because it contains an instantiation of a unit whose body hasn't been compiled; AI-00506 allows this behavior.

--------#37----------

PM=PS BC3204C/D, BC3205C/D [not all tests in all disputes]

These tests check that instantiations of generic units with unconstrained types as generic actual parameters are illegal if the generic bodies contain uses of the types that require a constraint. However, the generic bodies are compiled after the units that contain the instantiations, and some implementations create a dependence of the instantiating units on the generic units as allowed by AI-00408 and AI-00506 such that the compilation of the generic bodies makes the instantiating units obsolete—no errors are detected. The processing of these tests was modified by re-compiling the obsolete units; all intended errors were then detected by the compiler.

Some implementations detect no errors during compilation, and AI-00256 allows this behavior because none of the units is illegal with respect to the units on which it depends. However, all errors must then be detected at link time.

CHAPTER 11, EXCEPTIONS

--------#38----------

PM=PS CB1010B

The petitioner pleaded an AI-00325 case for not raising STORAGE_ERROR under certain circumstances due to peculiarities of the operating system. The FRT accepted the petitioner's dispute; the AVO ruled that the test must be passed by using an appropriate environment setting (and to use just one setting for all tests).

--------#39----------

EM=PS C34004C, C36204A

A petitioner argued that optimization may legitimately avoid the raising of some of these tests' expected exceptions. The AVO ruled that the tests may be graded as passed if only certain Report.Failed messages are output.

--------#40----------

TM=PS C34005P/S

These tests contain expressions of the form "I-X'FIRST+Y'FIRST", where X & Y are of an array type with a lower bound of INTEGER'FIRST; some implementations recognize that "-X'FIRST + Y'FIRST" is a loop invariant and so evaluate this part of the expression separately, which raises NUMERIC_ERROR. These tests were modified by inserting

parens at lines 187 & 262/263, respectively, to defeat the optimization and force the subtraction to be evaluated first.

--------#41----------

EM=NA C34007P/S

These tests include a check that the evaluation of the selector "all" raises CONSTRAINT_ERROR when the value of the object is null. Some implementations determine the result of the equality tests at lines 207 and 223, respectively, based on the subtype of the object; thus, the selector is not evaluated and no exception is raised, as allowed by Standard 11.6:7. The tests were graded passed given that their only Report.Failed output was the message "NO EXCEPTION FOR NULL.ALL - 2".

--------#42----------

EM=NA C41401A

This test checks that the evaluation of attribute prefixes that denote variables of an access type raises CONSTRAINT_ERROR when the value of the variable is null and the attribute is appropriate for an array or task type. Some implementations derive the array attribute values from the subtype, and thus the prefix is not evaluated and no exception is raised, for the checks at lines 77, 87, 97, 108, 121, 131, 141, 152, 165, & 175. This behavior was considered to be allowed by Standard 11.6:7.

--------#43----------

TM=PS C52008B

This test contains a sequence of three statements in which the third is intended to raise CONSTRAINT_ERROR and checks within the handler for this exception verify that the first and second statements were executed. One petitioner argued that the Standard 11.6:7&11 should be considered to allow the subtype check of the third statement (line 64) to be made in advance of the two preceding statements, such that the exception will be raised prior to their execution (and thus Report.Failed will be called from within the handler). The FRT concurred in the petitioner's opinion, and the AVO proposed a Test Modification that allowed the test to be passed. But no actual validation effort was made with this modification as this dispute was raised principally for information purposes (for the construction of an optimizer that wouldn't be invoked during validation).

--------#44----------

RJ*** C52005F

The petitioner argued that the initialization of a variable within a block's declarative part may be optimized away because the first reference to that same variable within the block's sequence of statements is an assignment to it. The FRT was undecided on this issue, and the AVO rejected the dispute as being too far from what was clearly allowed by the Standard 11.6. It was noted that the use of the variable in the block's exception handler might follow only the initialization, because an exception might be raised immediately before the first executable statement (as allowed by Standard 11.6:11).

--------#45----------

PM=PS C36204A, C36305A, C38202A, C45614A, C64103A/B, C64104A/N, C94001E/F, CB4006A, CC3125C, CC3305A..D, CE3704C, CE3804F/P

These tests each perform operations intended to raise CONSTRAINT_ERROR; but in each case the operation is part of either an explicit or implicit assignment of a value to a variable that is not later used. One compiler's optimization processing recognizes that the values will not be used and so eliminates the operations, thus avoiding the exceptions. The FRT concluded that this optimization was within the intent of Standard 11.6:7. The AVO ruled that the tests must be processed both with and without optimization, with the passed results from the unoptimized processing being the justification for the passed grade. (The entire ACVC was processed with optimization and this set of tests was also processed without optimization.)

CHAPTER 13, REPRESENTATION CLAUSES AND IMPLEMENTATION-DEPENDENT FEATURES

--------#46----------

RJ*** CD1009A/I, CD1C03A, CD2A-2**/-31*/-5**

One petitioner challenged these tests' requirement that 'SIZE length clauses be supported for integer, enumeration, and fixed-point types; this dispute was rejected, as the ARG had agreed that such clauses must be supported.

--------#47----------

RJ*** CD2A5**, ED2A56A [perhaps subsets of this series]

Some petitioners challenged these tests' requirement that the size length clause be supported (for the minimal size, as determined by the ARG) for fixed-point types; these disputes were rejected based on the ARG resolution.

--------#48----------

WD--- C32203A

This test implies that the sizes of a derived type and similarly constrained subtype of the parent type must be the same, but such a requirement is not stated by the Standard or by any Commentary.

--------#49----------

EM=NA CD1009E/F

These tests use length clauses for array types with type INTEGER components that specify the size to be 'LENGTH * INTEGER'SIZE. For Cray implementations, INTEGER'SIZE is 46 bits and SYSTEM.STORAGE_UNIT is 64 bits (a machine word) hence, the specified representation cannot be met without some of the array components crossing word boundaries. AI-00556 addresses the issue of support for size length clauses for array types; this Commentary has not been approved, and it does not conflict with the implementations' behavior.

--------#50----------

EM=PS C34009D/J

These tests check that 'SIZE for a composite type is greater than or equal to the sum of its components' 'SIZE values. This issue is the topic of Commentary AI-00825, which has not been considered; there is not an obvious interpretation. Some implementations represent array components, whose length depends on a discriminant with a default value, by implicit pointers into the heap space; thus, the 'SIZE of such a record type might be less than the sum of its components 'SIZEs, since the size of the heap space that is used by the varying-length array components is not counted. These tests were graded passed given that the only Report.Failed output was "INCORRECT 'BASE'SIZE", from line 195 of C34009D and line 193 of C34009J.

--------#51----------

RJ*** CD2A91A..E

The petitioner's challenge of these tests indicated a confusion between the length clauses for 'SIZE versus 'STORAGE_SIZE—the petitioner seemed to use former as the latter. The AVO rejected the dispute on the basis of Standard 9.1:1 & 9.2:2, the definition of the "value of a task object"; the petitioner seemed to interpret a task object as the code and data region that is associated with a task, rather than the access to that region.

--------#52----------

EM=NA CD2A53A

Several petitioners challenged this test's requirement that implementations support decimal 'small values (for fixed-point types). The FRT was divided on this point, although two approved Commentaries imply that such support is not mandatory.

--------#53----------

RJ*** C46051B, C55B16A

One petitioner challenged these tests' use of an enumeration representation clause that specified (some) negative values. The AVO rejected this dispute based on the ARG's draft of AI-00564 (which addresses this issue) and the fact that all prior implementations had passed these tests.

--------#54----------

EM=NA BD4006A

This test checks that non-static values in component and alignment clauses are rejected, but static alignment values of 8, 16, & 32 are assumed to be supported. Implementations are not required to support such alignments.

--------#55----------

TM=PS BD4007A, BD4009A, CD4051C

These tests use record representation clauses that place a component at a 0 offset from the start of a record with discriminants; but one implementation reserves a prefix of at least 1 bit at the head of any record type with discriminants, and so rejects the specified representation. The tests were modified by inserting '1' to change the offsets to from 0 to 10.

--------#56----------

WD--- BD4008A

The petitioners argued that this test should be withdrawn because it tested a point that was under deliberation by the ARG and not at all clear; the FRT concurred in the petitioner's complaint.

--------#57----------

RJ*** CD4061A

One petitioner argued that the implementation always allocated a storage for record types in 16-bit amounts, and so rejected this test's 'SIZE length clause's specified value of 5 (bits); the implementation would not allow packing of objects of such a type in less than 16 bits. This dispute was rejected on the basis of AI-00553/01.

--------#58----------

RJ*** CD5003A..CD5014Z (45 tests)

EM=NA

These 45 tests check the use of address clauses; a few petitioners argued that address clauses are meaningless in a virtual environment. The FRT did not agree with the challenges, citing the importance of interfacing to objects that are defined externally. In the case of the Rational Environment, only the Ada language is supported and so the usual concerns of interfacing to foreign language objects are irrelevant; in this case the failure to support address clauses was allowed.

--------#59----------

EM=NA B91001H

This test checks that an address clause for an entry cannot precede that or any other entry of the task. This implementation does not support interrupts, and so rejects any address clause for an entry, regardless of placement.

--------#60----------

EM=NA C86001F

WD---

This test checks that a test-defined package SYSTEM may be compiled and that it replaces the predefined package SYSTEM. This test was ruled inapplicable to some implementations which did not allow such re-compilation of SYSTEM, based on prior FRT

opinion that issue was undecided; it was later withdrawn, upon consideration of Commentary AI-00041, on which no action has been taken.

--------#61----------

EM=NA C92005B

This test includes a conversion of the 'STORAGE_SIZE value for a task to type INTEGER. For one implementation, that value exceeds INTEGER'LAST and the conversion raises CONSTRAINT_ERROR, which terminates the test.

--------#62----------

TM=PS A99007A

This test assigns a task's 'STORAGE_SIZE value to an INTEGER object. For one implementation, that assignment raises an exception because the value is greater than INTEGER'LAST. The test was modified by re-defining type INTEGER at line 13 with "TYPE INTEGER IS RANGE 0..MAX_INT; ".

--------#63----------

EM=PS CC1220A

This test evaluates the address of the same generic formal object of mode "in" at lines 35 and 66; it expects that address to be the same. The issues of what the address of a constant is, and whether it may change over the life of the object, are unclear; the AVO ruled that one implementation's behavior of returning different addresses at different places, for a constant, is acceptable pending resolution of AI-00203 by the ARG. The test was graded as passed even though the test reported "FAILED", given that the only call to Report.Failed occurred for the controversial check at line 66, which output the message "IMPROPER VALUE FOR 'ADDRESS".

--------#64----------

EM=PS CD1C04E

This test checks that a record representation clause for a derived type determines the values of the 'POSITION, 'FIRST_BIT, & 'LAST_BIT attributes of components of objects of the type, when the parent type has been given a different representation. For this implementation, SYSTEM.STORAGE_UNIT is 1, and thus 'FIRST_BIT always returns 0 and the check for inequality fails. The AVO ruled that this was acceptable behavior; the test was graded passed because all other checks were applicable and passed.

--------#65----------

TM=PS AD7203B

One petitioner presented a dispute for an implementation that, due to the allocation of storage for task stacks, would raise STORAGE_ERROR when this test is run. The test was modified by adding a 'STORAGE_SIZE length clause for the task type TSK at line 165 to specify an allocation of 1024 storage units for the activation of each task of the type.

--------#66----------

WD--- AD7206A

This test checks the use of the ADDRESS attribute for objects of many types, including constants; a petitioner challenged whether the attribute was well defined for constants, and the FRT agreed that it was not.

--------#67----------

PM=PS AD9001B, AD9004A

TM=PS AD9001B

These tests check that various subprograms may be interfaced to external routines (and hence have no Ada bodies). Some implementations require that a file specification exists for the interfaced foreign subprogram bodies. For these implementations a command was issued to the Librarian to inform it that the foreign bodies will be supplied at link time (the bodies are not actually needed by the program, so this command alone is sufficient).

For some other implementations, actual foreign bodies are required to exist; the test was processed in an environment that contained foreign bodies.

Another implementation rejected the use of a formal parameter of OUT mode for an interfaced subprogram; for this implementation, AD9001B was modified by commenting out line 26 (one of several interfaced subprograms).

--------#68----------

Q?... ED9002A

The petitioner included this test in a challenge on the use of the pragma INTERFACE. The AVO questioned how the challenge affected this test, and argued that it should not; the petitioner agreed, and the dispute was dropped.

--------#69----------

EM=NA CDA201C

This test instantiates Unchecked_Conversion with an array type with a non-static index constraint; some implementations do not support instantiations of Unchecked_Conversion with unconstrained types and so reject them. The AVO ruled that various restrictions on Unchecked_Conversion may be accepted for validation under ACVC 1.11, because AI-00590, which addresses issues involving Unchecked_Conversion, did not show an ARG consensus at the time of ACVC 1.11's release.

--------#70----------

EM=PS CD1009A, CD1009I, CD1C03A, CD2A21B/C, CD2A22J, CD2A23B, CD2A23A, CD2A24A, CD2A31A..C [various subsets are disputed for same reason]

These tests use instantiations of the support procedure Length_Check which uses Unchecked_Conversion according to the interpretation given in AI-00590. The AVO ruled that this interpretation is not binding under ACVC 1.11; the tests are ruled to be passed if they produce Failed messages only from the instances of Length_Check.

It was additionally noted that CD2A21C could print a Report.Failed message prior to the Report.Test message, as a consequence of allowable elaboration ordering of package bodies; such output was accepted under the ruling above.

--------#71----------

EM=PS CD1009M/V/W, CD1C04D, CD3014C/F, CD3015C/E/F/H/K, CD3022A

The petitioner argued that in the case where no length clause has specified the size of enumeration types, the effects of Unchecked_Conversion are not well defined. This point was not accepted by the FRT, but the dispute was accepted by the AVO for the same reasons as disputes concerning tests that used the support procedure Length_Check: the ARG deliberations on AI-00590, which addresses the effects of Unchecked_Conversion, were not begun until after ACVC 1.11 was released, and the public version of the Commentary (i.e., what is available from the on-line AJPO files) did not even hint at the interpretation assumed by the tests, and Standard 13.10.2 is ambiguous.

--------#72----------

TM=PS ENUMCHEK & LENCHECK

The generic support procedures, Length_Check or Enum_Check (in support files LENCHECK & ENUMCHEK), use the generic procedure Unchecked_Conversion. Some implementations reject instantiations of Unchecked_Conversion with array types that have non-static index constraints. The AVO ruled that since this issue was not addressed by AI-00590, which recommends required support for Unchecked_Conversion, and since AI-00590 is considered not binding under ACVC 1.11, the support procedures could be modified so as to remove their use of Unchecked_Conversion. Lines 40..43, 50, and 56..58 in LENCHECK and lines 42, 43, & 58..63 in ENUMCHEK were commented out.

CHAPTER 14, INPUT-OUTPUT

--------#73----------

EM=NA CE2103A/B, CE3107A

These tests abort with an unhandled exception when USE_ERROR is raised on the attempt to create an external file. This is acceptable behavior for those implementations that do not support external files as per AI-00332.

--------#74----------

TM=PS CE2103C/D

These tests close an empty file; however, for some IBM VM/SP HPO (CMS) implementations, the operating system does not allow an empty file to exist, and so the file is deleted and USE_ERROR is raised. The AVO ruled that this behavior is acceptable, given the operating system (cf. AI-00325), and that the tests be modified by inserting the following write statement into the tests at lines 56 and 55, respectively: "WRITE (TEST_FILE_ONE, 'A'); ".

--------#75----------

EM=PS CE2102C/H, CE2103A/B, CE3102B, & CE3107A

One implementation accepts all strings as legal file names, and thus these tests' checks for proper behavior on the use of illegal file names could not be made. The tests were graded as passed, given that the only Report. Failed output came from checks where a file name was expected to be illegal and all other checks were applicable and passed.

--------#76----------

TM=PS CE2108A..D, CE3112A/B

These tests check that temporary files are not accessible after the completion of the program that creates them; CE2108A, CE2108C, & CE3112A each create temporary files, and CE2108B, CE2108D, & CE3112B check that those respective files are not accessible. However, these latter tests also create temporary files. This implementation gives the same names to the temporary files in both the earlier- and later-processed tests of each pair; thus, CE2108B, CE2108D, & CE3112B report failed, as though they have accessed the earlier-created files. The second tests of each pair were modified to remove the code that created the (later) temporary file: lines 45..64 were commented out in CE2108B & CE2108D; lines 40..48 were commented out in CE3112B.

--------#77----------

EM=NA CE2108B/D/F/H, CE3112B/D

One implementation (a cross compiler to a bare target) simulated files on a bare target; these simulated files do not exist after the program that created them terminates. These tests all check the contents of files that were created and written to by earlier, partner tests; they were ruled to be inapplicable to this implementation because all files are temporary.

--------#78----------

WD--- CE2117A/B, CE3116A

Petitioners challenged the tests' attempt to use as a file name a string with an embedded space; the FRT concurred in the petitioners' dispute as well in the AVO's assertion that these tests lacked merit.

--------#79----------

EM=NA CE3202A

This test applies function NAME to the standard input file, which in some implementations has no name and thus USE_ERROR is raised but not handled, causing the test to abort. The AVO ruled that this behavior is acceptable pending any resolution of the issue by the ARG.

--------#80----------

RJ*** CE2102K

The petitioner challenged the test's requirement that a DIRECT_IO CLOSE not truncate the file; the petitioner argued that CLOSE should truncate the file, just as it does for SEQUENTIAL_IO & TEXT_IO. However, the FRT stated that such behavior was not intended by the Standard, which expected normal semantics for direct I/O—it is not normal to truncate the file on CLOSE.

--------#81----------

TM=PS CE2203A, CE2403A

EM=NA CE2203A

These tests check that, if an implementation can restrict the capacity of a file and an attempt to exceed that capacity is made, then USE_ERROR is raised; but they require that the capacity can be limited to 4096 characters or less. For one implementation, direct files could restricted but sequential files could not, yet the two tests use the same macro for the form parameter that is to effect the capacity restriction. In this case, the form parameter was chosen so as to enable the direct I/O test (CE2403A) to pass, although the output for CE2203A was thus misleading.

In another dispute, the implementations could restrict file capacity only by device tracks, which gave an effective capacity of approximately 50K bytes. For these implementations, the tests' output loops were modified so as to attempt to exceed that much larger capacity.

--------#82----------

EM=NA CE3106A/B & CHECKFILE

These tests check TEXT_IO operations; they do not allow an implementation to pad a text line with trailing blanks. One petitioner argued that if an implementation has record-oriented I/O where the smallest record size is one word, that the last record of a line should be padded with blanks (to complete the word). As this behavior is explicitly permitted by the ACVC support procedure CHECKFILE, this implementation's behavior was accepted as conforming pending an interpretation by the ARG.

--------#83----------

EM=NA CE3111B, CE3115A

The tests assume that output from one internal file is unbuffered and may be immediately read by another file that shares the same external file. Some implementations buffer output and thus raise END_ERROR on the attempts to read at lines 87 & 101, respectively.

--------#84----------

EM=PS EE3301B, EE3405B, EE3410F

These tests check certain I/O operations on the current default output file, including standard output. One IBM implementation outputs the ASCII form-feed character which has no effect on the standard IBM output devices; in general, there is no common form-feed mechanism for the devices. Thus, the printed output from this test did not contain the expected page breaks. The AVO ruled that these tests should be considered passed if none of the tests' internal checks was failed (i.e., if the tests report "TENTATIVELY PASSED").

--------#85----------

EM=NA CE3413B

This test checks that TEXT_IO.PAGE raised LAYOUT_ERROR if the page number exceeds COUNT'LAST. The test makes uses the expression "COUNT'LAST > 150000" as an applicability check, and is applicable only when the expression is TRUE (to avoid excessive processing demands for time and disk space). But for some implementations, the expression can raise CONSTRAINT_ERROR on the implicit conversion of the literal '150000' to type COUNT; since there is no handler for this exception, test execution will terminate. The AVO ruled that this test be graded inapplicable because it checks certain file operations which were not supported anyway.

For another implementation, there was a low limit on the number of PAGE operations that could be given for a single file (approximately 1800), and so DEVICE_ERROR was raised and caused the test to terminate. This implementation limitation was ruled to be acceptable as per AI-00325, and the test was ruled to be inapplicable.

--------#86----------

NOTE CE3602A

One petitioner remarked that this test contained several mistaken calls to TEXT_IO.SET_LINE_LENGTH, although these did not affect the correctness or the processing of the test. The AVO informed the ACVC Reviewers and the ACVC Team.

--------#87----------

TM=PS CE3605A

This test attempts to write a line with 516 characters; this exceeds some implementations' default output-line limit, and thus USE_ERROR is raised. As this behavior is allowed by AI-00534 (although there is no ACVC test to check that truly unbounded lines are supported—cf. AI-00534), the test was modified so as to reduce the amount of text output to within the system's limits.

--------#88----------

EM=NA CE3806G

TM=PS CE3901A

These tests expect that implementations that do not support external files will raise USE_ERROR on the attempt to create a file at line 52; but AI-00332 permits implementations to raise NAME_ERROR. CE3901A was modified by inserting '|NAME_ERROR' into the exception choice at line 52; CE3806G was simply graded inapplicable when it terminates with an unhandled exception.

Appendix B.

PRECIS OF FAST-REACTION NOTICES ISSUED FOR ACVC 1.11 TESTS DURING FISCAL YEAR 1991

This appendix presents precis of all of the dispute deliberations conducted by IDA with its consulting body of Ada experts, called the Fast Reaction Team (FRT). These deliberations are initiated by an e-mail message from IDA that describes the dispute; this message is called a Fast Reaction Notice (FRN). Each FRN is labelled according to its date of issue; this label is in the form "FRN <date>.<sequence_letter>" (a sequence letter is used to distinguish FRNs issued on the same date). The entire file of e-mail deliberations on an FRN is also referred to by the FRN label.

These precis make reference to the Ada standard, the Ada Commentaries, and to particular ACVC tests; the references use the following forms, respectively:

Standard <chapter>.<section>.<subsection>:<paragraph> (e.g. "Standard 3.5:4"),

AI-<number>/<version> (e.g. "AI-00301/07"), and

test <test_name> (e.g. "test C52008B").

Also, the precis often refer to an ACVC report procedure, viz. Report.Failed; this procedure is invoked from executable tests when a check is failed.

frn901003 MAY COMPONENT CLAUSES BE GIVEN FOR OBJECTS OF NON-STATIC SUBTYPES?

The issue is whether a component clause may be given for a component with a non-static subtype with static constraints; this issue is addressed in the unresolved (un-voted on) Commentary AI-00301/07. Although AI-301 as written currently interprets the Standard contrary to the petitioner's challenge, the FRT supported the petitioner's position that there should not be a test for the unresolved issue. The AVO withdrew test BD4008A.

frn901005: TESTING HOW MANY FILES CAN BE CREATED IS WITHOUT MERIT

The issue is whether an implementation may reject filenames that contain embedded blanks (an accidental quality of the test code) and whether the test objectives have merit. The FRT concurred in both the petitioner's complaint as well as Lehman's opinion that no merit existed for the tests' objectives; the AVO withdrew tests CE2117A & B (and, later, CE3116A).

frn901009: CONVERSION OF <LARGE_FLOAT>'SAFE_LARGE TO <SMALL> MAY SUCCEED

The issue is whether an implementation whose floating-point types differ only in mantissa may fail to raise an exception on the conversion of a larger (range) floating-point type's 'SAFE_LARGE to a smaller type. The FRT agreed that such a conversion need not raise an exception in this case. The AVO ruled that tests C64103A & C95084A may be graded passed by the Evaluation Modification of allowing the relevant Report.Failed output. (Hilfinger asserts that Ada 9X will likely NOT allow the implementation's behavior).

frn901011: REQUIREMENTS FOR PREMATURE USE OF A DEFERRED CONSTANT LACK MERIT

The issue is whether the use of a deferred constant prior to its full declaration must either raise PROGRAM_ERROR or result in predictable behavior. The tests uses a deferred constant as the initialization expression in a default discriminant value; an object of the record type is declared prior to private part. Must the value that is assigned in an object declaration be the same as that which is assigned in the constant's later full declaration? The FRT concurred in the petitioner's arguments, and added some of their own. The AVO withdrew test C74308A.

frn901018: MAY THE 'ADDRESS OF A GENERIC "IN" PARAMETER VARY?

The issue is whether the address of a generic IN parameter may vary in the course of program execution. Test CC1220A contains a generic formal parameter of type SYSTEM.ADDRESS that is initialized by default to the address of another formal parameter of mode IN; within the generic body, the address attribute is again applied to the formal parameter and it is expected to equal the earlier-assigned default value. For the petitioner's implementation, the address has changed. The FRT agreed that, although the implementation's behavior seemed strange, the ACVC should not require the 'ADDRESS attribute to return the same value in these cases. The AVO allowed the implementation to pass validation with the exhibited behavior; the test was changed for ACVC 1.12, and no longer contains this problem.

frn901105: COMPILATION CAPACITY IS EXCEEDED—TOO MANY PAGE FAULTS?

The issue is whether it is acceptable for an implementation to claim a capacity limitation for tests C85006A..E—compilation requires over 1E6 page faults! The FRT variously did not like the implied implementation limitation, but generally agreed that the test should be split and processed (vs. rejecting the dispute). (Lehman & Dewar continued with lengthy arguments re the purpose of validation.) The AVO ruled that the tests must be split and passed.

frn901112: WHAT IS THE "HUMAN READABLE FORM" OF TEXT_IO PAGE BREAKS?

The issue is whether the Standard requires that TEXT_IO page breaks be manifest as actual new pages on an implementation's printer. Cohen gives an excellent and compelling discussion of the general issue in his first 21 November response; others of the FRT opposed the implementation's limitation. The AVO concurred in Cohen's caution against requiring that tests EE3301B, EE3405B, & EE3410F be passed as is, and ruled that their expected grading by inspection of printer output be waived for this implementation (the tests also make self-grading checks, which were passed).

frn901211: MAY TEXT_IO PAD OUTPUT WITH TRAILING BLANKS?

The issue is whether an implementation "may legally append blanks to the end of any line" (support subprogram CHECKFILE contains the quoted text). The FRT was divided on how strongly to press for implementing what most agreed to be Ada semantics—i.e., no padding with trailing blanks. It was argued that forcing such conformity to the Standard could make the use of TEXT_IO on the implementation problematic, as the resulting files would not be what the other system tools expected. It was also suggested that the implementation should offer a mode of operation that supported strict Standard semantics. The AVO ruled that test CE3106A & B may be graded NA (certainly, CHECKFILE alone provides a basis for this ruling). The petitioner was advised of the FRT's comments, which hint that there might later be a requirement for no-blanks behavior from the ARG.

frn901219: FUNCTION EXPRESSION IN CASE ALTERNATIVE IS TREATED AS TYPE

The issue is a challenge to Standard 5.4:3-4, which together imply that when a case expression is a function call all values of the type must be represented in the alternatives,

not merely those of even a static subtype. The FRT recommended that this issue be forwarded to Tucker Taft for consideration re Ada 9X. The AVO rejected the dispute and ruled that test C54A13D is correct.

frn910117: SUPPORT OF SIZE LENGTH CLAUSES FOR FIXED-POINT (CF frn900131.b)

The issue is whether there is any likelihood of a change to ARG thinking on size length clauses for fixed-point types (frn900131.b addressed the issue also;—here it is asked whether Ada 9X will affect the requirements). Dewar (and Goodenough, orally) responded that the requirements of the tests were supported by the ARG. The AVO rejected the dispute, and advised the petitioner of the effects that would be expected in future ACVC versions (which happen to not be made under ACVC 1.11). The CD2A5* tests were required to be passed.

frn910123: LIMITATION ON NUMBER OF TASK OBJECTS DUE TO OS THREADS LIMIT

The issue is whether an implementation that uses operating-system threads for tasks may do so even if the OS does not provide a sufficient number of threads; also, what is "a sufficient number"? The issue proved to be contentious and unclear in resolution, since ruling that some implementation limit is unacceptable seldom finds any supporting guidance from the Standard. Although there was much discussion, the issue remained unclear; the AVO made no ruling, because the petitioner withdrew the dispute and chose to use a version of the operating system that caused no problem for test A98002A.

frn910214: ALLOWABLE OPTIMIZATION (IN C34004C, C36204A, & C52005F)

The issue is What checks may be optimized away by dead-variable removal and copy propagation. The issues of allowable optimization are known to be difficult to resolve; the FRT deliberations on this petitioner's cases proved easy for some of the cases, but difficult for another. The AVO rejected the optimization that was requested for test C52005F as being too far from what was clearly allowed (Ploedereder's opinion on this was a basis); the AVO allowed tests C34004C & C36204A to be passed by Evaluation Modification.

frn910312: OPERATING SYSTEM ABORTS PROGRAM IN SOME CASES WHERE STORAGE_ERROR IS EXPECTED

The issue is whether this implementation is justified (in AI-325 terms) in failing to raise STORAGE_ERROR when heap space is exceeded. Extensive questioning of the petitioner showed that the implementation could not ensure that programs would always raise an exception when storage limits were exceeded; this is because the OS imposes some limits on total storage usage that might be exceeded before the limit on some particular storage (e.g., the heap) was reached--upon which the OS would abort the program. The FRT agreed that the petitioner's case was an acceptable AI-325 justification for its failure, in certain conditions, of test CB1010B. The AVO required the implementation to process the entire ACVC with limits on heap and other storage that prevented the OS abortion from occurring.

frn910412: VARYING STACK SIZES TO ENABLE SOME TESTS TO BE PASSED

The issue is whether it was appropriate for an implementation to process the some of the ACVC tests with different options than what are used for all other tests: in particular, the petitioner requests changing the sizes of the primary and task stacks for 30 or 31 tests for each of two implementations, respectively. The AVO cited Pro90 5.2.1, which specifies that single set of options will be used for validation—the reason for this dispute. The FRT was satisfied with the implementation's behavior and the proposed changes to the default stack sizes. The AVO allowed the implementation to be processed with any needed changes to the stack sizes. (Later correspondence indicated that fewer tests would need changes, and ultimately only test C32107A used different sizes.)

frn910502.a: MAY PARAMETER EVALUATION BE OMITTED IF THE PARAMETER ISN'T USED?

The issue is whether a formal parameter may be considered "dead" if it is not used within the subprogram, so that the actual parameter need not be evaluated. The actual case is a type conversion that is intended to raise CONSTRAINT_ERROR. The FRT agreed that the petitioner's optimization is permissible; the Standard 11.6(3 & 7) and Presentation Commentary AI-00168 were cited in support of the dispute. Lehman noted that other tests of the C64103* series were similarly vulnerable to optimization but not disputed. The AVO ruled that test C64103A may be graded passed with an Evaluation Modification to allow the two Report.Failed messages and a "FAILED" result.

frn910502.b: WHAT DOES 'SIZE MEAN FOR UNCONSTRAINED TYPES?

The issue is essentially that of Commentary AI-00825: what is the meaning of the SIZE attribute for unconstrained types? The petitioner asserts that the implementation allocates dynamic array components in the heap and thus maintains only pointers to them within a record; 'SIZE for the record will include only the size of the pointers and other components, not the heap space. The FRT agreed that 'SIZE in this case should not be tested. The AVO ruled that tests C34009D & J may be graded passed by an Evaluation Modification that allows Report.Failed messages for the affected checks of 'SIZE and (of course) a "FAILED" result.

frn910703.a: RE-ORDERING OF "ASSIGNMENT STATEMENTS"

The issue was whether the subtype check entailed in an assignment statement (at least, if not the assignment itself) could be re-ordered relative to other statements. It was remarked that the difference between advancing the evaluation of an expression (which might raise an exception) versus the subtype check for an assignment was not obvious, except in the rules of the Standard—i.e., the latter seems no more dangerous or surprising than the former, which is explicitly allowed by Standard 11.6(11). The consensus was that the re-ordering should be allowed. The AVO made only a tentative ruling, as there was no validation at stake and it was not clear to what extent this allowed re-ordering would affect the ACVC; only test C52008B was cited in the dispute, for which the AVO suggested a Test Modification of commenting out certain lines.

frn910904: DOES AI-506 APPLY TO SINGLE COMPILATION UNITS? (NO)

The issue is whether AI-00506 may be applied to a single compilation unit that contains an instantiation of a generic unit prior to the compilation of its body, which occurs later within the unit. Goodenough confirmed that AI-506 applies only to separately compiled units. The AVO rejected the petitioner's dispute of tests CC1305B, BC3204B, & BC3205B.

frn910922: OPTIMIZING AWAY "ISOLATED" SUBTYPE CHECKS

The issue is whether an implementation may apply Standard 11.6(7) optimization even in cases where there is no predefined operation but simply a subtype check (e.g., the subtype check for the input value read by TEXT_IO.GET if the actual parameter is not subsequently used). The consensus was that these checks may be optimized away. The AVO ruled that the 19 affected tests must be processed both with and without optimization,

and graded as passed based on the unoptimized results (i.e., the entire ACVC was processed with optimization, and the affected tests were processed additionally without optimization).

Appendix C.

A SAMPLE OF FAST REACTION TEAM DELIBERATION

This appendix contains examples of the exchanges among Fast-Reaction Team members in deliberation of a dispute. Although the presented dialogue is from an FRN from before FY 1991, this sample is chosen because of the brevity of the dispute and responses, the change in opinion due to IDA's interaction, and the nature of the dispute—a question of debatable legality which was resolved in favor of the petitioner on urgings of caution from the FRT.

Subject: FRN 891216-b: CONVERTED EXPRESSION IS NOT STATIC?

FRTeam:

The following dispute has been received for ACVC 1.11 tests C46022A.

---Dan

---

reference: C46022A, 6.6(1)

summary: Explicit conversion operands are not static.

test code:

```
26          IF FLOAT5     (4_722_348_468_471_135_731_712) /=
27                        4_722_348_468_471_135_731_712.0 THEN
28              FAILED ("INCORRECT VALUE (1)");
29          END IF;
30      EXCEPTION
31          WHEN NUMERIC_ERROR | CONSTRAINT_ERROR =>
32              FAILED ("NUMERIC/CONSTRAINT_ERROR RAISED FOR " &
33              "CONVERSION (1)");
34          WHEN OTHERS =>
35              FAILED ("UNEXPECTED EXCEPTION RAISED FOR " &
36                  "CONVERSION (1)");
```

discussion:

An explicit conversion is not a static expression (see AI-00703/00 [which I cannot find, John]) and thus is evaluated dynamically. The evaluation of an explicit conversion evaluates the expression given as the operand (6.6/1). This evaluation is done at run time and therefore the expression must fit within the largetst predefined integer type, in this case. This test should be withdrawn.

::::::::::

From dewar@acf2.NYU.EDU Mon Dec 18 03:21:10 1989

Seems that the implementor is right here, and I would prefer the test to be withdrawn, rather than merely marked as inapplicable for the particular implementor.

::::::::::

From jbg@SEI.CMU.EDU Mon Dec 18 10:18:56 1989

The implementer is correct. The conversion of the integer literal to the largest integer type (other than universal_integer) is allowed to raise an exception.

[AI-703 is a presentation AI, and so is in the ai-presentation.all file.]

::::::::::

From knapper@ida.org Mon Dec 18 14:29:07 1989

The implementer is correct here.

Bob K

::::::::::

From lehman@ida.org Mon Dec 18 20:09:53 1989

FRTeam:

I'm unsure of the basis for the allowance of an exception here:

the case is "FLOAT (<universal_integer_expression>)";

4.6(2nd"15"—under Notes) says "no implicit conversion is ever applied to the operand of an explicit type conversion";

4.9 says that a numeric literal is static (and thus the value must be returned without exception);

so it would seem that the operand of the explicit conversion is static;

BUT

explicit conversions are not static;

4.6(1) says that the evaluation of an explicit conversion "evaluates the expression given as the operand" (making any 4.9 claims to staticness null and void?)

So although no explicit implicit conversion may be made (4.6(15.2)), there is an implicit implicit conversion due to the dynamic nature of conversion!

Dan

From jbg@SEI.CMU.EDU Tue Dec 19 10:23:24 1989

Good point, Dan! 4.10(5) says:

> For the evaluation of an operation of a nonstatic universal expression, an implementation is allowed to raise the exception NUMERIC_ERROR only if the result of the operation (or an operand; see AI-00181) is ... an integer value greater than SYSTEM.MAX_INT or less than SYSTEM.MIN_INT.

It is clear that FLOAT(14234123423212343212343 2) is not a universal expression, so this paragraph does not apply. It is also clear that the operand _is_ a static expression, so this paragraph does not apply. So one could legalistically argue that the conversion must not raise an exception, as long as the operand lies within the range of FLOAT.

Even if the target type is user-defined with nonstatic bounds, there is no legalistic justification for raising an exception on the basis that the operand value cannot be represented at run-time. The implementation must convert the operand value to the target type (presumably at compile time, at which point an exception can be raised if the result value is not in the range of the target base type) and then check the bounds at run-time.

So I think it is an interesting ramification that the test is correct.

From dewar@acf2.NYU.EDU Tue Dec 19 12:30:53 1989

> I have never quite understood the position on what is and what is not a static expression. It seems clear to me that whether an expression is a syntactic issue, determined by Appendix-E, rather than a semantic condition which can apply to any expressions. Following this line of reasoning, I really can't see why the operand of the expression should be regarded as static.
>
> I would tread a little carefully on this subject. The idea that the compiler is forced to do this conversion at compile time seems peculiar to me, given that the conversion ITSELF is clearly non-static. It is true that compilers SHOULD compute this sort of thing at compile time, but in terms of the model of the language, it seems right that the compiler should only be forced to compute static expressions at compile time.
>
> Furthermore, I am aware of at least one compiler (the NYU compiler) which makes the assumption that ONLY static expressions need be computed at compile time, and I have the impression that other compilers have the same approach to the world.

From ploedere@tartan.com Tue Jan 2 14:32:18 1990

> The issue is sufficiently unclear (in my reading of the exchanges) that the tests should be withdrawn pending an ARG ruling.
>
> I concur with Robert that it would be strange if the language semantics required a compile-time evaluation for a non-static expression, i.e., a type conversion (or, alternatively, a run-time universal_integer computation).

Erhard

Appendix D.

A SAMPLE OF TEST, PROCESSING, & EVALUATION MODIFICATIONS

Among the dispute-resolution mechanisms that the AVO employs is the issuance of Test, Processing, or Evaluation Modifications for tests affected by a dispute. A principal motivation in requiring one of these Modifications, as opposed to withdrawing the affected tests, is to ensure that the ACVC is applied as uniformly as practical to all validated implementations, and to retain disputed tests in the ACVC if they can serve validation. This appendix presents a sample of each of these Modifications.

CA2009C and CA2009F were graded inapplicable by Evaluation Modification as directed by the AVO. These tests contain instantiations of a generic unit prior to the separate compilation of that unit's body. As allowed by AI-257, the compilation of the generic unit bodies is rejected.

CE3804H was graded passed by Evaluation Modification as directed by the AVO. This test requires that the string "-3.525" can be read from a file using FLOAT_IO and that an equality comparison with the numeric literal '-3.525' will evaluate to TRUE; however, because -3.525 is not a model number, this comparison may evaluate to FALSE (Ada standard 4.9:12). This implementation's compile-time and run-time evaluation algorithms differ; thus, this check for equality fails and Report.Failed is called at line 81, which outputs the message "WIDTH CHARACTERS NOT READ". All other checks were passed.

BC3204C and BC3205D were graded passed by Processing Modification as directed by the AVO. These tests check that instantiations of generic units with unconstrained types as generic actual parameters are illegal if the generic bodies contain uses of the types that require a constraint. However, the generic bodies are compiled after the units that contain the instantiations, and this implementation creates a dependence of the instantiating units on the generic units as allowed by AI-00408 and AI-00506 such that the compilation of the generic bodies makes the instantiating units obsolete—no errors are detected. The processing of these tests was modified by re-compiling the obsolete units; all intended errors were then detected by the compiler.

CE3901A was graded passed by Test Modification as directed by the AVO. This test expects that implementations that do not support external files will raise USE_ERROR on the attempt to create a file at line 52; this implementation raises NAME_ERROR, as allowed by AI-00332. The test was modified by inserting '| NAME_ERROR' into the exception choice at line 52, and the modified test was passed.

Appendix E.

IDA LOG FILE FOR VALIDATION ACTIONS

This appendix presents a sample of IDA's internal log file of validation actions. This file enables IDA to assess the performance of the component organizations of the Ada certification body in fulfilling their respective duties. This file is also one of the sources of the information that is presented at the end of each week to all members of the Ada certification body in the AVO's Status Report (see Appendix F).

For each Ada validation, IDA records the completion date for key actions in the validation process (dates are in the form *yymmdd*). Each of these actions is explained below, along with the log file's column heading under which the completion date is recorded. The "Draft" entry is usually the initial entry, but occasionally the "DoC/NoC" entry is first (e.g., validation "n222").

- Draft (from the AVF)—IDA's receipt of the draft VSR from the AVF
- Cmmnts (to the AVF)—IDA's critical "Comments" to the AVF re the VSR
- DoC/NoC (from the AVF)—"Declaration of Conformance"/"Notice of completion", these are required items for IDA's issuance of a certificate request; they are often contained in the VSR delivery
- VC Req. (to the AIC, cc AVF)—"Validation Certificate Request", IDA's request to the AJPO to issue a certificate for the validation
- Final ["in"] (from the AVF)—IDA's receipt of the final VSR from the AVF
- FCmmnts (to the AVF)—"Final Comments", IDA's critical comments to the AVF re the final VSR (optional, mainly when earlier comments are mis-applied)
- Final ["OUT"] (to the AJPO)—IDA's delivery of the final VSR to the AJPO

The log file's first three columns contain reference information about the validation. These are explained below:

- Seq#--this is a combination of an AVF indicator and the three sequence digits of the certificate number, which IDA assigns; the AVF indicators are: "a" for AFNOR, "i" for IABG, "n" for NCC, "s" for NIST, and "w" for WPAFB
- AVF#—this is an AVF reference number; in the case of WPAFB, the full reference number is a combination of a contract reference and a VSR reference (sequence) number; these VSR numbers are only listed for group validations with a common contract number (e.g., "w180 910426gse" followed by "w181 - 480" etc.)
- Cstmr.—this is an abbreviated designation of the AVF customer

Finally, an "*" indicates that there are notes regarding the particular action (these are appended to the log file--not shown in this appendix). E.g., the asterisks under the "VC Req." column for w181, w183, w185, and w189 are for a note explaining that these certificates were later corrected and re-issued.

Seq# AVF# (abbreviated entries)	Cstmr.	Draft in	Cmmnts. OUT	DoC/NoC in	VC Req. OUT	Final in	FCmmnts OUT	Final OUT
w180 910426gse	GSE mbh	910709	910730	910716	910725	910821	910827	
w181 " -480	"	"	"	"	"	"	"	
w182 " -481	"	"	"	"	"	"	"	
w183 " -482	"	"	"	"	"	"	"	
w184 " -483	"	"	"	"	"	"	"	
w185 " -484	"	"	"	"	"	"	"	
w186 " -485	"	"	"	"	"	"	"	
w187 " -486	"	"	"	"	"	"	"	
w188 " -487	"	"	"	"	"	"	"	
w189 " -488	"	"	"	"	"	"	"	
w190 " -489	"	"	"	"	"	"	"	
s191 90act525_1	IntrAct	910714	"	in vsr	910731	910806	na	910815
s192 " _2		"	"	"	"	"	"	910814
l193 VSR-097	Alsys-G	910715	910803	"	"	911017	na	911025
l194 VSR-089	TeleSft	910729	"	"	910805	910826	na	910906
w195 910422als	Alsys-I	910807+	910814*	910812	910816	910926	na	911010
w196 " -500	"	"	"	"	"	"	na	"
w197 " -501	"	"	"	"	"	"	na	"
n198 90502/78	NChinal	910829	910911	910909	910911*			
n199 90502/79	SD-Scl.	910916	911004	910912	910930			
w200 910716vrx	Verdix	"	911008	910923	910929			
w201 " -491	"	"	"	"	"			
w202 " -492	"	"	"	"	"			
w203 " -493	"	"	"	"	"			
w204 " -494	"	"	"	"	"			
w205 " -495	"	"	"	"	"			
w206 " -496	"	"	"	"	911008*			
w207 " -497	"	"	"	"	910929			
w208 " -498	"	"	"	"	"			
w209 " -502	"	"	"	"	"			
w210 " -503	"	"	"	"	"			
w211 " -504	"	"	"	"	"			
w212 " -505	"	"	"	"	"			
w213 " -510	"	"	"	"	"			
w214 " -511	"	"	"	"	911008*			
w215 " -512	"	"	"	"	911008*			
s216 90nec525_1	NEC	910918	911011	in vsr	911001	911024	na	911025
s217 " _2	"	"	"	"	911009	"	na	"
w218 910815mss	Mridian	911003	911105	911007	911020			
w219 " -513	"	"	"	"	"			
w220 " -514	"	"	"	"	"			
w221 " -515	"	"	"	"	"			
n222 90502/31	IC Ltd	911008		911004	911020			
w223 910730tel	Cray R.	"		911007*	911029			
w224 910621alt	Altech	911007		911015	911030			
w225 " -507	"	"		"	"			
s226 90dec530_1	DEC	911029	911105	in vsr	"	911107	911108	911108
w227 910904hp	H-P Co.	911031	911108	911107	911107			
w228 " -517	"	"	"	"	"			
l229 VSR-098	TeleSft	911114		in vsr	911115			
n230 90502/80	SD-Scl.	911118						

Appendix F.

IDA STATUS REPORT TO THE SPONSOR AND THE ADA VALIDATION FACILITIES

At the end of each week, IDA issues by e-mail to the sponsor and the five Ada Validation Facilities a Status Report on validation actions. This report provides status information on the following items: implementer disputes and registration requests; the outstanding actions, and the dates of completed actions, for current validation efforts; and pertinent e-mail that has been received or sent by IDA. The Status Report also serves to ensure that no lost communications are undetected for more than one week (i.e., the Status Report is expected by all on Monday, and it will indicate by omission any unreceived communications of the preceding week).

The following page presents a Status Report that is abbreviated by the truncation of the information for four of the five Ada Validation Facilities (the truncated sections are similar to the section for National Computing Centre (NCC) which is retained).

From Lehman@ida.org Fri Aug 23 21:48:00 1991
To: AVF-Managers@ajpo.sei.cmu.edu Cc: RogersD@ajpo.sei.cmu.edu
Subject: STATUS 91-08-23

AVO WEEKLY STATUS REPORT
for the week ending 91-08-23

DISPUTES QUEUE
Resolved:
910820.NCC B38101A & compiler error placement (@line 49 vice 65)

Need more information:

Pending:
910821.WPAFB E28002B/5D & PRAGMA LIST
CA2009A & D & instantiations precede bodies
AD7203B & memory exhaustion???--for THIS!???

REGISTRATION QUEUE
Done:

Held for more information:
91-08-15 NIST B161#90uni515--UNISYS 2200/600 & relatives

Pending:
91-08-23 WPAFB B002#900115vrx--VERDIX
" B028/9#900625har--HARRIS
" B101#900925vrx--VERDIX
" B149/52/4/6#910318--VERDIX

GENERAL E-MAIL
08/16 dlehman@ajpo.sei. STATUS 91-08-16

CORRESPONDENCE AND VALIDATION STATUS FOR EACH AVF

In general, the next action belongs to leftmost empty-column's owner.
>>>> for NCC ::
08/20 nccotd!ncc-avf@re Comment on B38101A <<Dear Dan, In checking the p
08/20 lehman@ida.org AVO RE B83101A--OK <<Dear Jon: You are correct i

Seq#	AVF#	Draft	Cmnts.	DoC	VC Req.	Final	FCmnts	2ndFinal	Deliv.
n134	90502/73	910318	910718	in vsr	910320				

>>>> for NIST :::

[... --similar sections are repeated for each of the five AVFs]

Distribution List for IDA Document D-1111

NAME AND ADDRESS	NUMBER OF COPIES
Sponsor	
Dr. John Solomond Ada Joint Program Office Room 3E114, The Pentagon Washington, DC 20301-3081	5
Other	
Defense Technical I[illegible]ion Center Cameron Station Alexandria, VA 22314	2
Ms. Chris Anderson Air Force Armament Lab Eglin, AFB Florida 32542-5434	2
Dr. William Dashiell Ada Compiler Validation Manager Software Standards Validation Group National Bureau of Standards Gaitherburg, MD 20899	1
Mr. Bobby Evans Ada Validation Facility Standard Languages and Environments Div. Engineering Applications Directorate DCS/Communications-Computer Systems Wright-Patterson AFB Ohio, 45433-6503	1
Mr. Jon Leigh Ada Validation Facility Manager National Centre for Information Technology National Computing Centre Ltd. Oxford Road, Manchester M1 7ED England	1
Mr. Alphonse Philip AFNOR Tour Europe, Cedex 7 F-92080, Paris la Defence France	1

NAME AND ADDRESS	NUMBER OF COPIES
Mr. Michael Tonndorf IABG, Dept. ITE Einsteinstr. 20 W-8012 Ottobrunn Germany	1
IDA	
General Larry D. Welch, HQ	1
Mr. Philip L. Major, HQ	1
Dr. Robert E. Roberts, HQ	1
Ms. Ruth L. Greenstein, HQ	1
Dr. Richard J. Ivanetich, CSED	1
Ms. Anne Douville, CSED	1
Ms. Audrey A. Hook, CSED	5
Mr. R. Danford Lehman, CSED	5
Mr. Terry Mayfield, CSED	1
Ms. Sylvia W. Reynolds, CSED	2
Dr. Richard L. Wexelblat, CSED	1
IDA Control & Distribution Vault	3